



Sandia
National
Laboratories

Exceptional service in the national interest

ME469: Common Discretization Approaches: Edge-based, Vertex-Centered (EBVC), and Cell-Centered (CC)

Stefan P. Domino^{1,2}

¹ Computational Thermal and Fluid Mechanics, Sandia National Laboratories

² Institute for Computational and Mathematical Engineering, Stanford

This presentation has been authored by an employee of National Technology & Engineering Solutions of Sandia, LLC under Contract No. DE-NA0003525 with the U.S. Department of Energy (DOE). The employee owns all right, title and interest in and to the presentation and is solely responsible for its contents. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this article or allow others to do so, for United States Government purposes. The DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan.

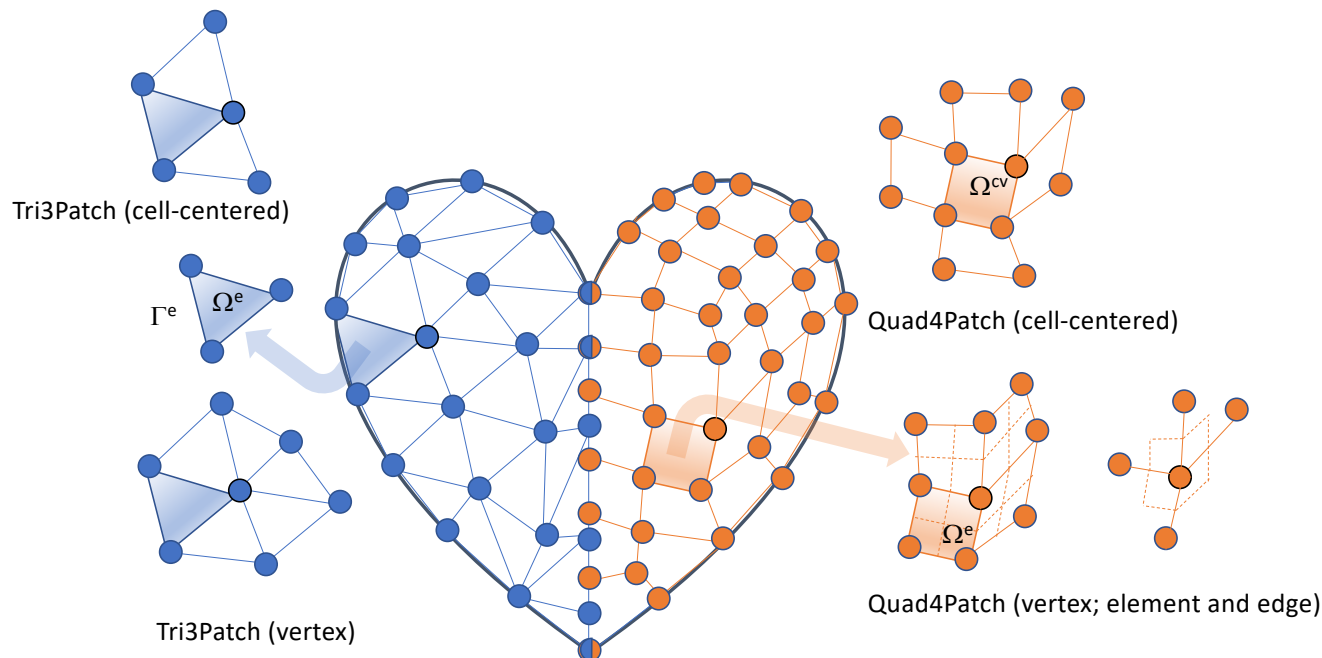
SAND2018-4536 PE





Review of Discretization Options: New, a nodal-basis...

- Degree-of-freedom (DOF) for:
 - Cell-centered: Stencil is based on a element:face:element
 - DOFs at vertices of elements, or “nodes”, element:node (CVFEM, FEM), edge:node (EBVC)



- Definition of an interpolation function:

$$\phi_{ip} = \sum_n N_n^{ip} \phi_n$$

- N_n^{ip} is the Lagrange function associated with node n
- ϕ_n is the value of the DOF at node n
- The nodal basis functions obey equipartition of unity and satisfy, $N_n^{x_j} = \delta_{nj}$



Fundamentals of Discretization: Surface vs Volume Integrations

- Given a partial differential equation (PDE) and associated volumetric form:

$$\int \frac{\partial F_j}{\partial x_j} dV = \int S dV$$

- Applying Gauss Divergence provides the standard finite volume form for fluxes in surface integral form:

$$\int \frac{\partial F_j}{\partial x_j} dV = \int F_j n_j dS \longrightarrow \int F_j n_j dS = \int S dV$$

- We can also multiple PDE by an arbitrary test function, w , and integrate over a volume,

$$\begin{aligned} \int w \frac{\partial F_j}{\partial x_j} dV &= \int w S dV \\ \int \frac{\partial w}{\partial x_j} F_j dV - \int w F_j dS &= \int w S dV \end{aligned}$$

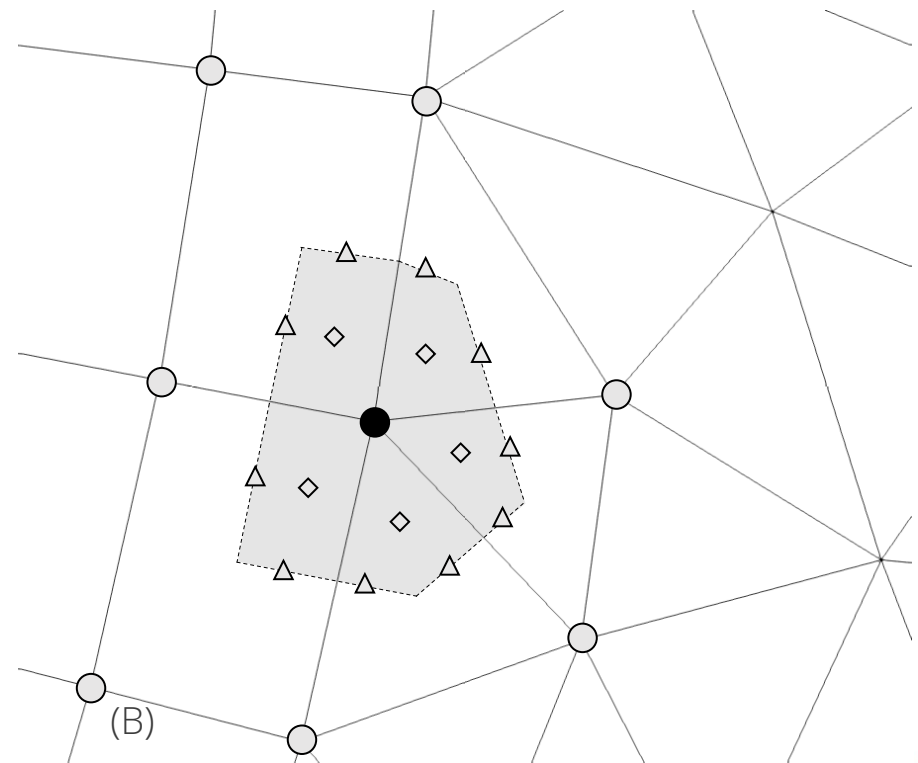
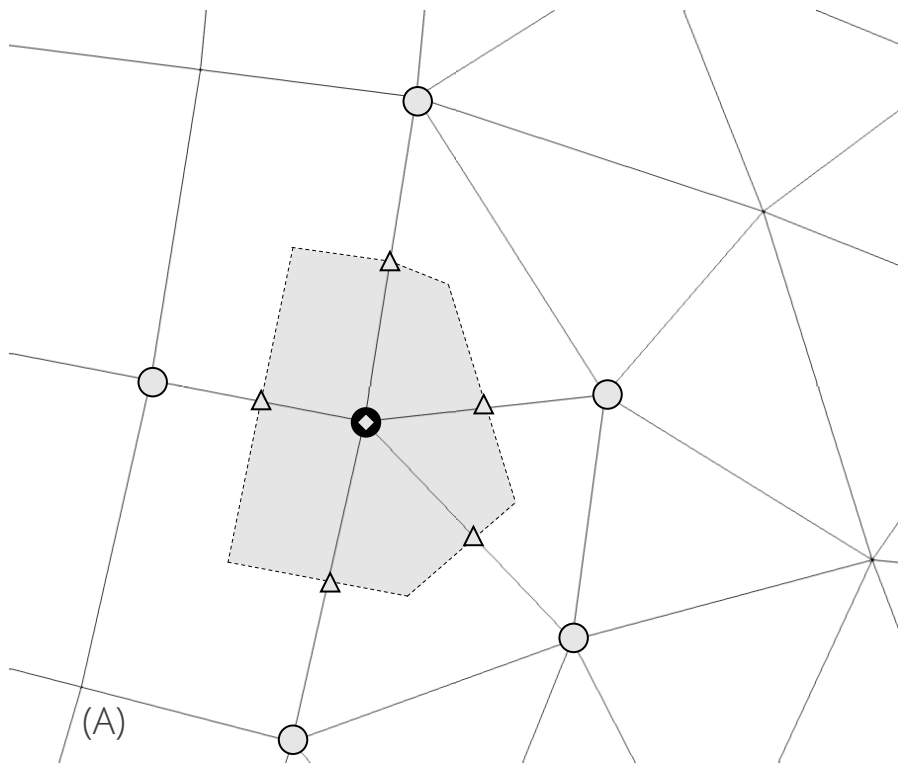
Next, integrate by parts and apply Gauss-Divergence. Note, that test function must be differentiable – shown here, at least once..

$$\frac{\partial w F_j}{\partial x_j} = w \frac{\partial F_j}{\partial x_j} + \frac{\partial w}{\partial x_j} F_j$$



Edge-Based Vertex-Centered Leverages the Dual-Volume Element-based Description of a Control Volume Finite Element Method (CVFEM)

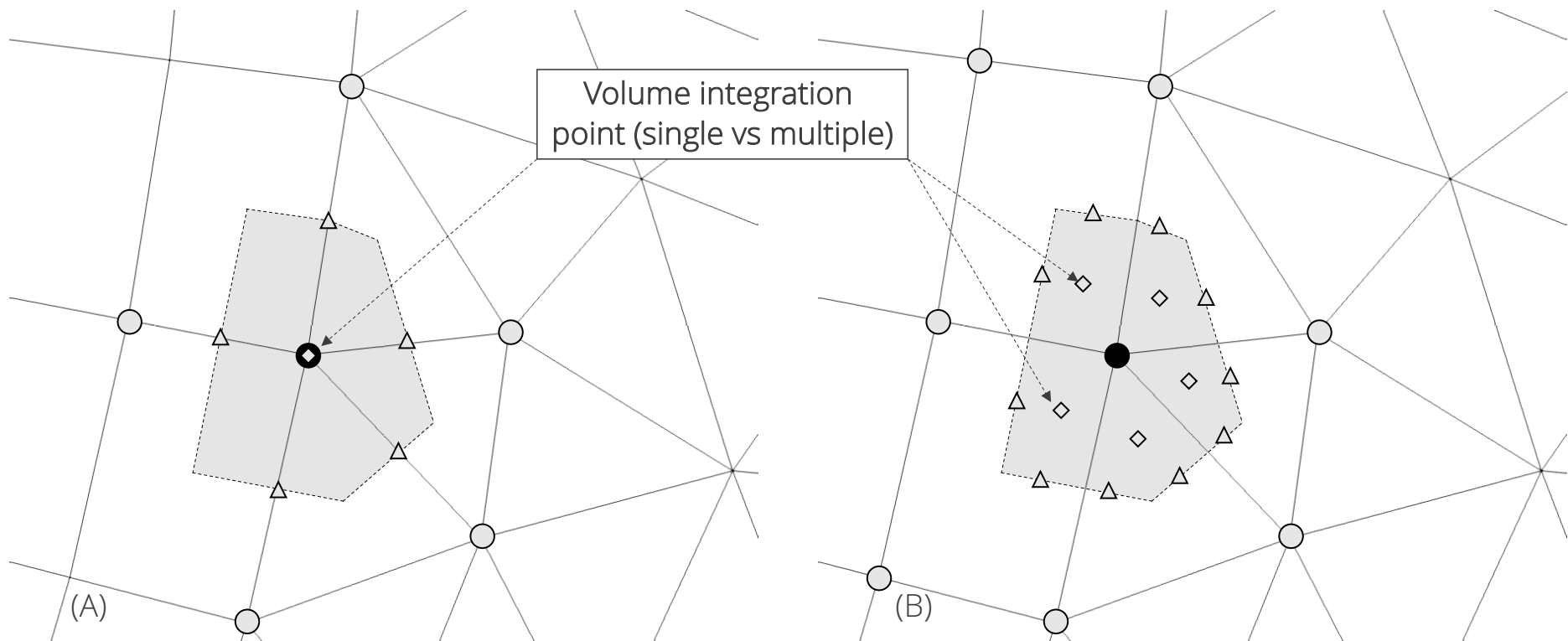
- EBVC (A) and CVFEM (B) – As shown below, the dual-volume and integration point layout is very similar





Edge-Based Vertex-Centered Leverages the Dual-Volume Element-based Description of a Control Volume Finite Element Method (CVFEM)

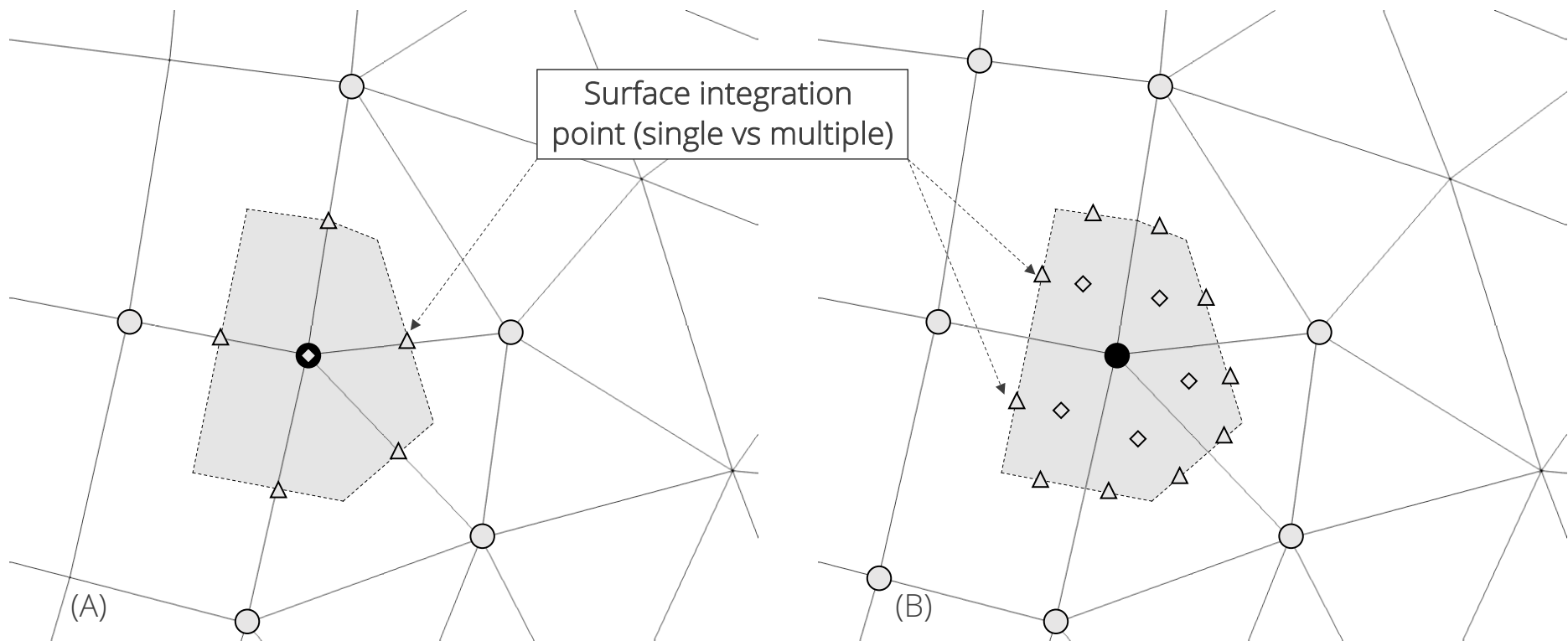
- EBVC (A) and CVFEM (B) – As shown below, the dual-volume and integration point layout is very similar





Edge-Based Vertex-Centered Leverages the Dual-Volume Element-based Description of a Control Volume Finite Element Method (CVFEM)

- EBVC (A) and CVFEM (B) – As shown below, the dual-volume and integration point layout is very similar





Deep Dive on CVFEM

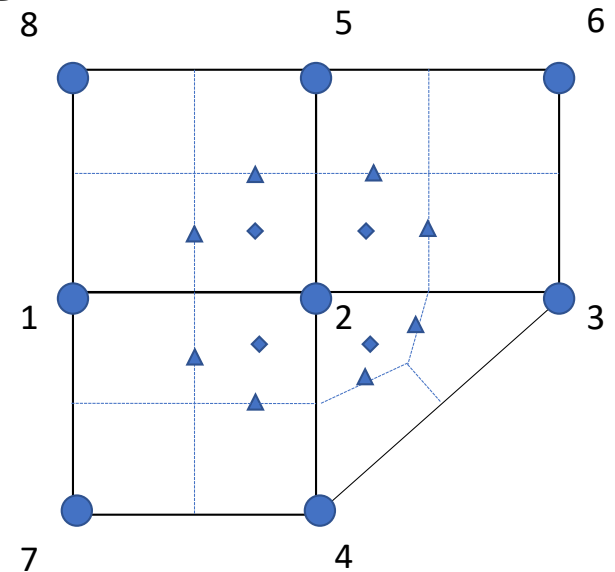
- CVFEM is a discretization scheme that:
 - Iterates over locally-owned elements for Time/Source/etc. (volumetric-based terms)
 - Iterates over locally-owned elements for Advection/Diffusion/etc. (integrated by parts terms)

Below is the patch of elements connected to node 2 (a global matrix row number)

- A *dual-volume* is defined within each element

Any value of the DOF within the element can be computed based on:

$$\phi_{ip} = \sum_n N_n^{ip} \phi_n$$

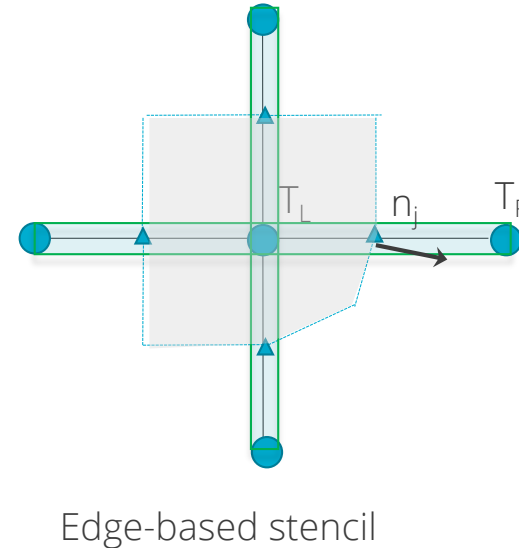
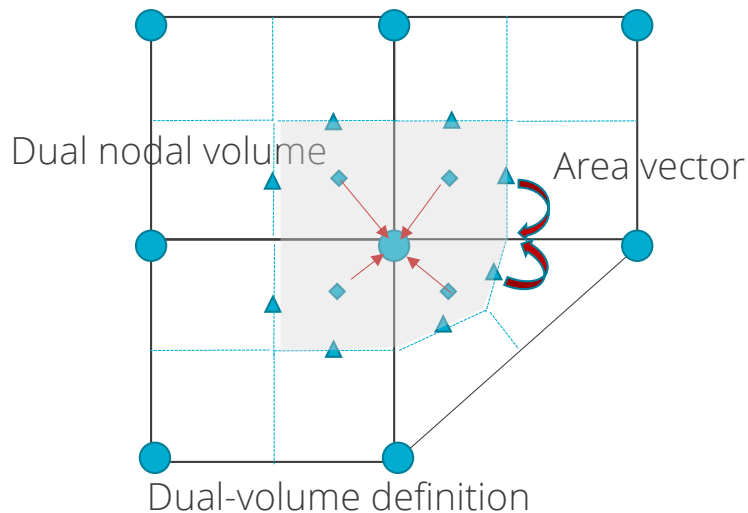


Sample patch of elements



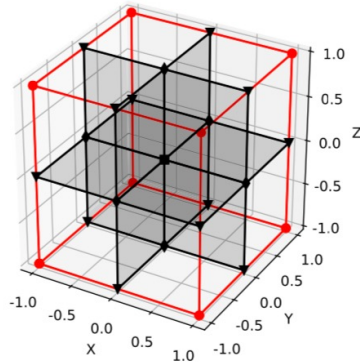
The Control Volume for EBVC is Defined by the **Dual-Volume**

- All primitives are collocated at the vertices of the elements with equal-order interpolation
- A dual mesh is constructed to obtain flux and volume quadrature locations
- Classic two-state, “L” and “R” approach provides spatially second-order accuracy
- Iterate **Nodes** for volume-based contributions
- Iterate **Edges** for surface-based contributions

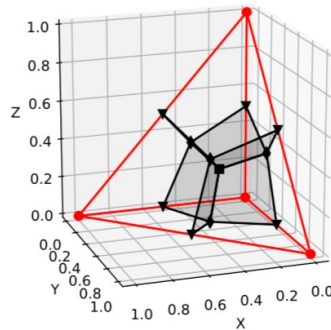




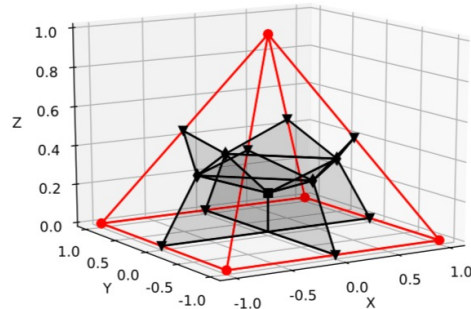
Dual Volume Definitions for Hybrid (Hex/Tet/Pyramid/Wedge) Meshes



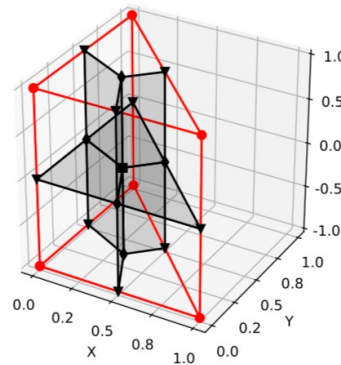
(a) Hexahedral topology (Hex8).



(b) Tetrahedral topology (Tet4).



(c) Pyramid topology (Pyramid5).



(d) Wedge topology (Wedge6).

Fig. 1. CVFEM element and dual-volume definition for the low-order topologies.

- Domino, et. al, "An assessment of atypical mesh topologies for low-Mach large-eddy simulation" 2019
- Generalized unstructured meshes support the ability to capture complex geometries, while minimizing the meshing time
- For example, near a solid wall, you might have a near-structured Hex-based mesh that is transitioned to unstructured
- For the mesh to be conformal, the faces for each adjacent element topology should match
- Drives Hex8 (Quad4 face) : Pyramid5 (Quad4 face); Pyramid5 (Tri3 face) to Tet4 (Tri3 face)
 - The Pyramid is known as a *transition* element

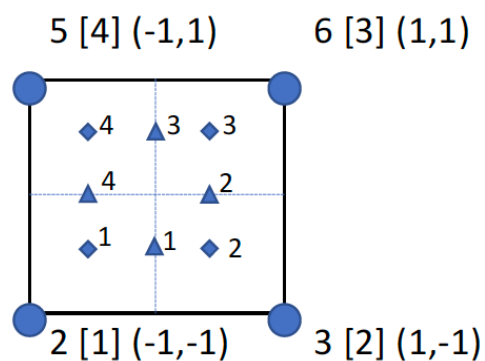


Deep Dive on CVFEM: Element-loops with Rich Basis

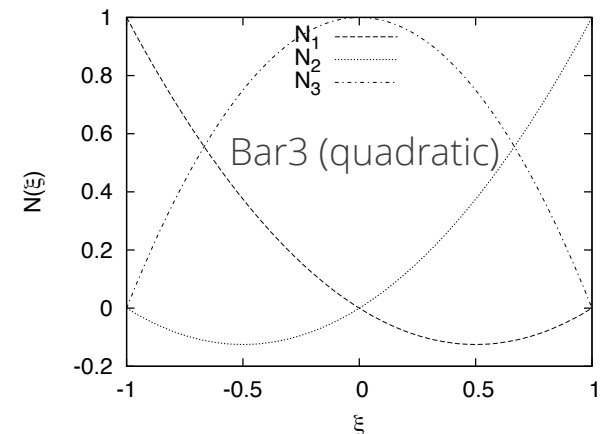
- For each element, recall that a dual volume has been constructed
- Volume-based contributions are evaluated at the sub-control volume integration points (diamonds)
- Surface-based contributions are evaluated at the sub-control surface integration points (triangles)

We define an isoparametric element that ranges from -1:1 in the ξ - (x-direction) and η - (y-direction) direction

Basis Functions for a Quad4



$$\begin{aligned} N_1 &= \frac{1}{4}(1 - \xi)(1 - \eta) \\ N_2 &= \frac{1}{4}(1 + \xi)(1 - \eta) \\ N_3 &= \frac{1}{4}(1 + \xi)(1 + \eta) \\ N_4 &= \frac{1}{4}(1 - \xi)(1 + \eta) \end{aligned}$$



For example, the isoparametric coordinate for ip 1 is (0.0, -0.5)



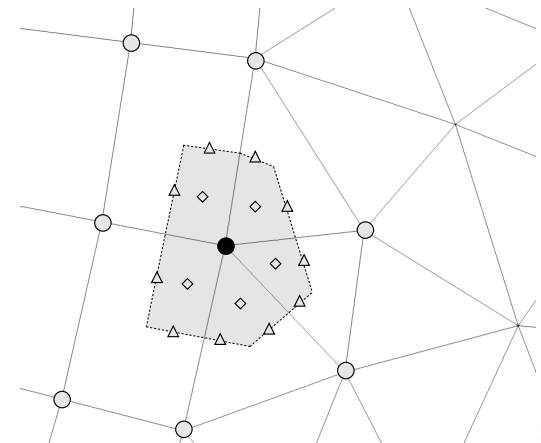
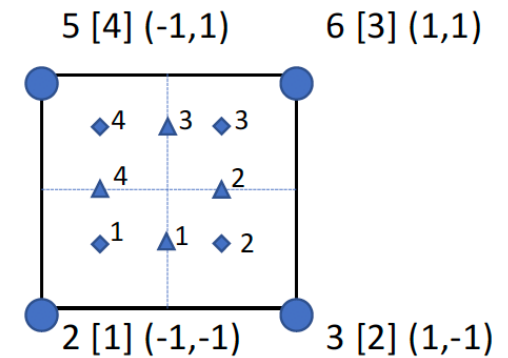
Deep Dive on CVFEM: Implicit Time Discretization

- Backward Euler (two state) and is first-order accurate (A-stable)
- BDF2 (three state) is second-order accurate (A-stable)
- This term is assembled over an element iteration and drives a consistent mass matrix with a full node:element:node connectivity

$$\int w \frac{\partial \rho \phi}{\partial t} dV \approx \sum_{ip} \frac{(\gamma_1 \rho_{ip}^{n+1} \phi_{ip}^{n+1} + \gamma_2 \rho_{ip}^n \phi_{ip}^n + \gamma_3 \rho_{ip}^{n-1} \phi_{ip}^{n-1})}{\Delta t} V_{ip}$$

- For uniform time stepping, the coefficients are give by,
 - $\gamma_1 = 3/2$ $\gamma_2 = -2$ $\gamma_3 = 1/2$
- You can easily see how the underlying basis pulls in the full node:element_node stencil:

$$\phi_{ip} = \sum_n N_n^{ip} \phi_n$$





Deep Dive on CVFEM: Implicit Time Discretization; Code

- <https://github.com/NaluCFD/Nalu/blob/master/src/kernel/ScalarMassElemKernel.C>

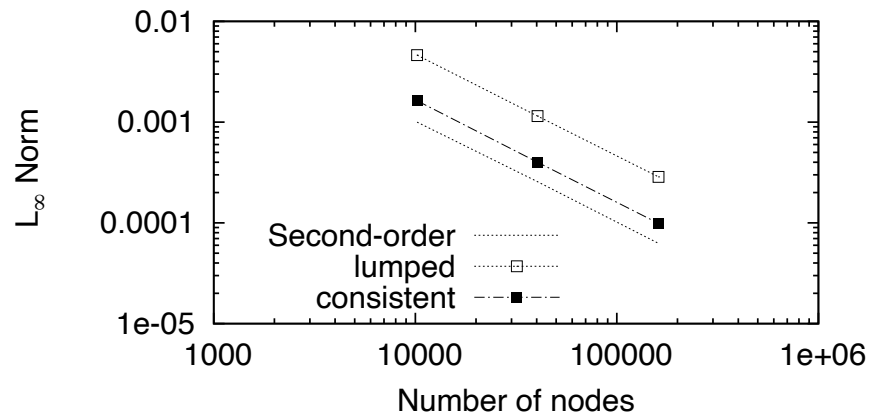


The Consistent Mass Matrix

- When evaluating the volumetric contributions using the full stencil, this approach is known as a *consistent mass matrix* as compared to the previous *lumped mass matrix* that is supported by CC and EBVC

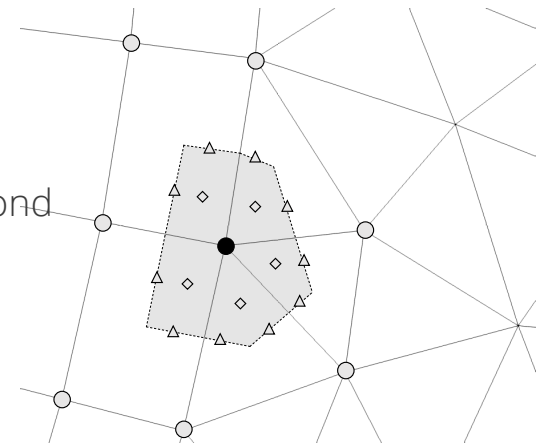
$$\int w \frac{\partial \rho \phi}{\partial t} dV \approx \sum_{ip} \frac{(\gamma_1 \rho_{ip}^{n+1} \phi_{ip}^{n+1} + \gamma_2 \rho_{ip}^n \phi_{ip}^n + \gamma_3 \rho_{ip}^{n-1} \phi_{ip}^{n-1})}{\Delta t} V_{ip} \quad (\text{Lumped})$$

$$\int w \frac{\partial \rho \phi}{\partial t} dV \approx \sum_{nd} \frac{(\gamma_1 \rho_{nd}^{n+1} \phi_{nd}^{n+1} + \gamma_2 \rho_{nd}^n \phi_{nd}^n + \gamma_3 \rho_{nd}^{n-1} \phi_{nd}^{n-1})}{\Delta t} V_{nd} \quad (\text{Consistent})$$



Convecting Taylor
Vortex analytical
solution; error
computed at 1 second

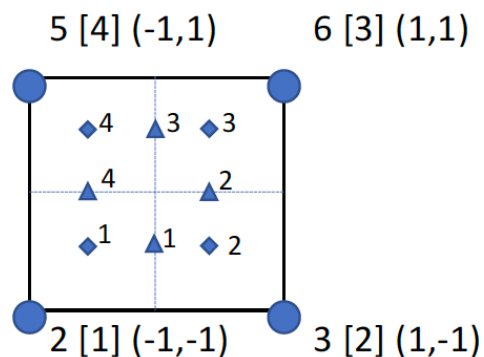
Shown to be the
convergence rate,
2nd order





Deep Dive on CVFEM: Source Term Discretization

- Source terms for CVFEM are also assembled over an element or nodal loop
- In some cases, the source term can be complex, i.e., includes gradients, which drives either a nodal assembly of these quantities to the nodes (recall, the projected nodal gradient), or local evaluation using the shape function derivatives (see upcoming diffusion operator)



$$\int w S^\phi dV \approx \sum_{ip} S_{ip}^\phi V_{ip}$$



Deep Dive on CVFEM: Source Term Discretization; Code

- https://github.com/NaluCFD/Nalu/blob/master/src/user_functions/SteadyThermal3dContactSrcElemKernel.C

```
template<typename AlgTraits>
void
SteadyThermal3dContactSrcElemKernel<AlgTraits>::execute(
    SharedMemView<DoubleType*>& /* lhs */,
    SharedMemView<DoubleType*>& rhs,
    ScratchViews<DoubleType>& scratchViews)
{
    // Forcing nDim = 3 instead of using AlgTraits::nDim_ here to avoid compiler
    // warnings when this template is instantiated for 2-D topologies.
    NALU_ALIGNED DoubleType w_scvCoords[3];

    SharedMemView<DoubleType*>& v_coordinates = scratchViews.get_scratch_view_2D(*coordinates_);
    SharedMemView<DoubleType*>& v_scv_volume = scratchViews.get_me_views(CURRENT_COORDINATES).scv_volume;

    // interpolate to ips and evaluate source
    for ( int ip = 0; ip < AlgTraits::numScvIp_; ++ip ) {

        // nearest node to ip
        const int nearestNode = ipNodeMap_[ip];

        // zero out
        for ( int j = 0; j < AlgTraits::nDim_; ++j )
            w_scvCoords[j] = 0.0;

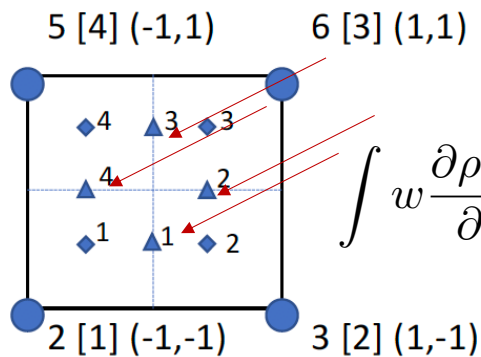
        for ( int ic = 0; ic < AlgTraits::nodesPerElement_; ++ic ) {
            const DoubleType r = v_shape_function_(ip,ic);
            for ( int j = 0; j < AlgTraits::nDim_; ++j )
                w_scvCoords[j] += r*v_coordinates(ic,j);
        }

        rhs(nearestNode) += k_/4.0*(2.0*a_pi_)*(2.0*a_pi_)*
            stk::math::cos(2.0*a_pi_* w_scvCoords[0])
            + stk::math::cos(2.0*a_pi_* w_scvCoords[1])
            + stk::math::cos(2.0*a_pi_* w_scvCoords[2]))*v_scv_volume(ip);
    }
}
```



Deep Dive on CVFEM: Advection Discretization (no stabilization)

- For advection, we have transformed the volume integral to a surface integration
- Therefore, a patch of elements are required for the full assembly at node 2



$$\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV \approx \sum_{ip} (\rho u_j)_{ip} \phi_{ip} n_j dS \approx \sum_{ip} \dot{m}_{ip} \phi_{ip}$$

Notes:

1. Common to integrate-by-parts, however, not required
2. Advection term need not be in divergence form (non-conserved form is suitable)


- Recall, that the mass flow rate at an integration point is prescribed
- Integration points can also be shifted from the sub-control surface to the edge midpoint (while still using the integration point area vector)
- This is a *central-* or *Galerkin-based* advection operator



Deep Dive on CVFEM: Advection Discretization Non-Conserved Form

- The advection term can be integrated by parts, or not; moreover, the PDE can drive a non-conservative equation form:

$$\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV = \int w \rho u_j \phi n_j dS - \int \rho u_j \phi \frac{\partial w}{\partial x_j} dV$$


$$\int w \rho u_j \frac{\partial \phi}{\partial x_j} dV + \int w \phi \frac{\partial \rho u_j}{\partial x_j} dV$$

$$\int w \frac{\partial \rho \phi}{\partial t} dV = \int w \rho \frac{\partial \phi}{\partial t} dV + \int w \phi \frac{\partial \rho}{\partial t} dV$$

- Non-conserved form, unlike CC and EBVC, provides no added complexity

$$\int w \left(\rho \frac{\partial \phi}{\partial t} + \rho u_j \frac{\partial \phi}{\partial x_j} \right) dV$$



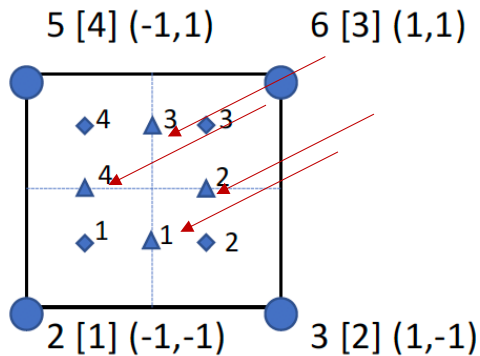
Deep Dive on CVFEM: Advection Discretization; Code

- <https://github.com/NaluCFD/Nalu/blob/master/src/kernel/ScalarAdvDiffElemKernel.C>
- This routine includes advection and diffusion
- Recall that integration point value is provided by nodal loop over the underlying nodal basis for this element
- Also note that this routine is valid for all types of supported elements – both low- and higher-order (polynomial promotion – coming later in the Quarter)



Deep Dive on CVFEM: Diffusion Discretization

- For diffusion, we have transformed the volume integral to a surface integration
- Therefore, a patch of elements are required for the full assembly at node 2
- Note that the CVFEM approach is absent any non-orthogonality corrections;
- However, high aspect ratio elements are now challenging...



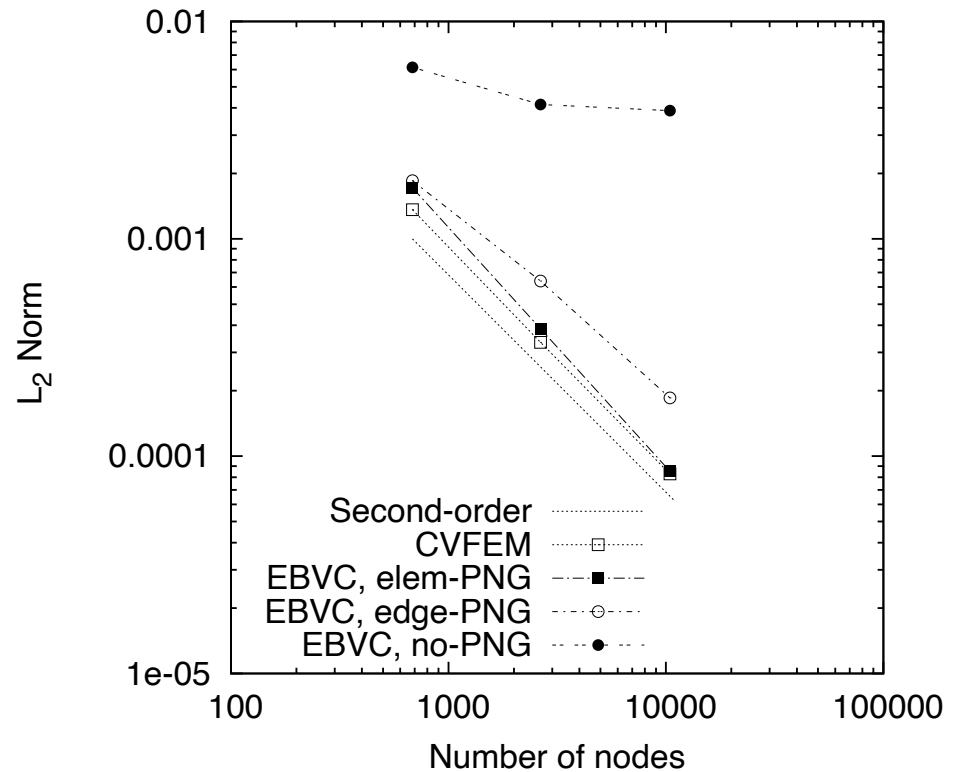
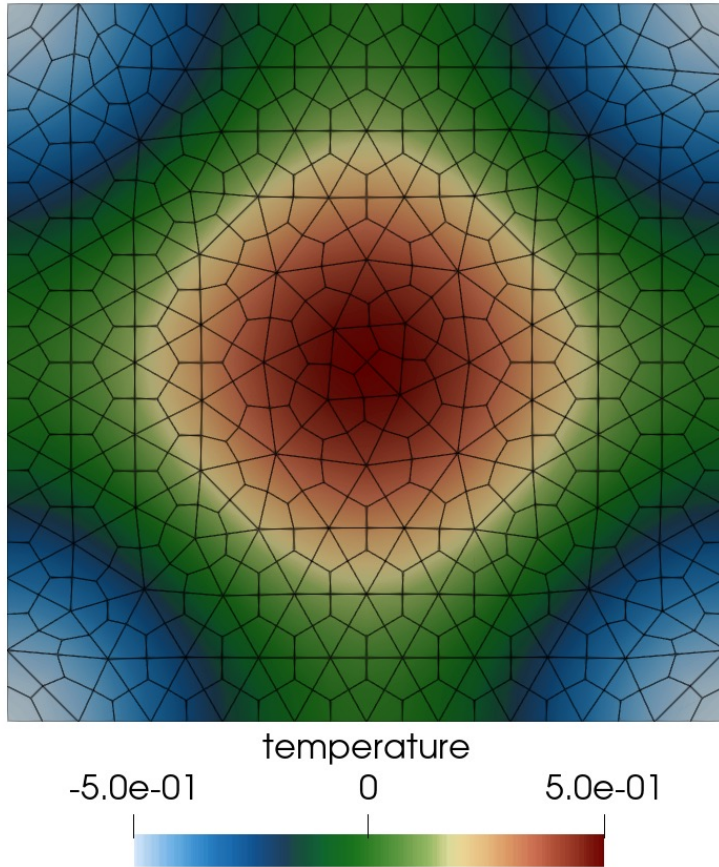
$$\begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{bmatrix} \frac{\partial N}{\partial x} \\ \frac{\partial N}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial N}{\partial \xi} \\ \frac{\partial N}{\partial \eta} \end{bmatrix} \quad \mathbf{J}^{-1} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}^{-1}$$

$$\begin{bmatrix} \frac{\partial N}{\partial x} \\ \frac{\partial N}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial N}{\partial \xi} \\ \frac{\partial N}{\partial \eta} \end{bmatrix}$$

$$\int w \frac{\partial q_j}{\partial x_j} dV \approx - \sum_{ip} \frac{\mu}{Sc_{ip}} \frac{\partial \phi}{\partial x_{j_{ip}}} n_j dS = - \sum_{ip} \frac{\mu}{Sc_{ip}} \sum_{nd} \frac{\partial N_{nd}^{ip}}{\partial x_j} \phi_{nd} A_j^{ip}$$



Deep Dive on CVFEM: Diffusion Discretization; Code





A Note on Conservation

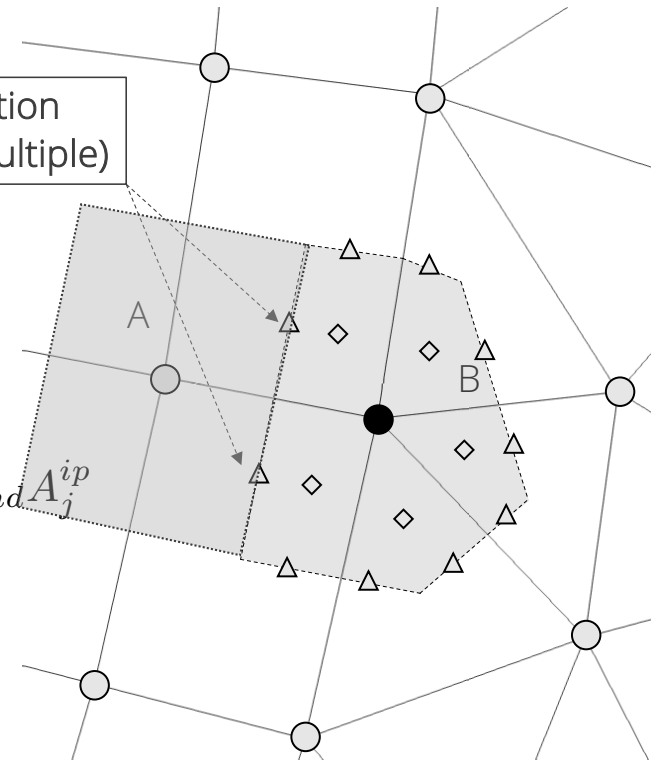
The CC, EBVC, and CVFEM all have one aspect in common.....

- Specifically, a flux contribution is evaluated at the control volume face, and dotted with the area surface normal vector
- This quantity, kg (stuff)/s, “leaves” control volume A, and fully enters control volume B and is *conserved*

Surface integration point (single vs multiple)

$$\int w \frac{\partial q_j}{\partial x_j} dV \approx - \sum_{ip} \frac{\mu}{S_{cip}} \frac{\partial \phi}{\partial x_j} n_j dS = - \sum_{ip} \frac{\mu}{S_{cip}} \sum_{nd} \frac{\partial N_{nd}^{ip}}{\partial x_j} \phi_{nd} A_j^{ip}$$

- For each integration point, the Left and Right nodal state is defined into which flux contributions are assembled
- No loss of mass/momentum/energy/etc.





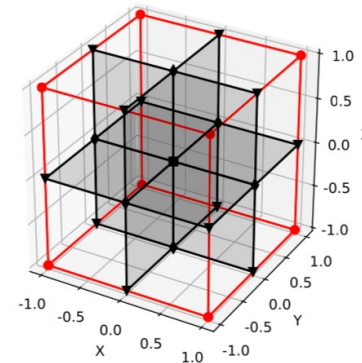
A Note on Polymorphism: Class Shape : Circle/Square/etc.

Recall, that the element type has been characterized by a set of unique attributes, take for example, a Hex8:

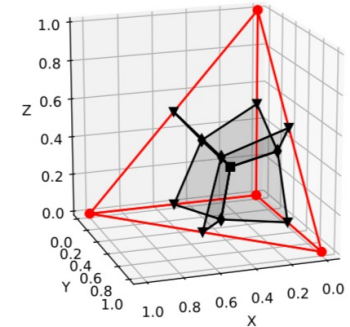
- Nodes Per Element: 8
- Number of surface integration points: 12
- Number of volume integration points: 8
- Number of Faces: 6
- Face Topology: Quad4
- Assembly Algorithms can be templated, or expecting an integration rule defined by the element type

Pseudo C++

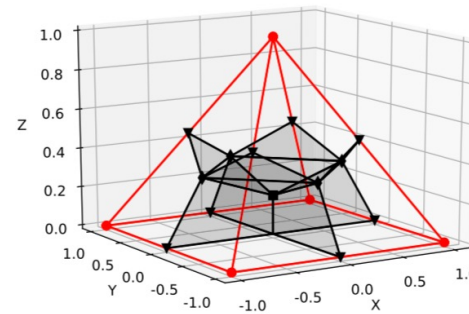
```
>Shape mySquare = new Square();  
>Shape myCircle = new Circle();  
>mySquare->volume();  
>myCircle->volume();
```



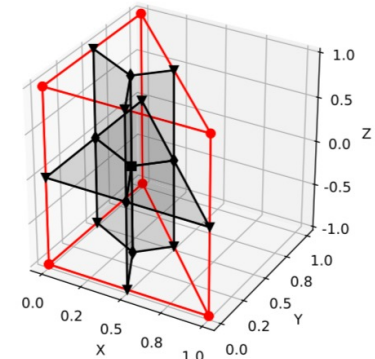
(a) Hexahedral topology (Hex8).



(b) Tetrahedral topology (Tet4).



(c) Pyramid topology (Pyramid5).



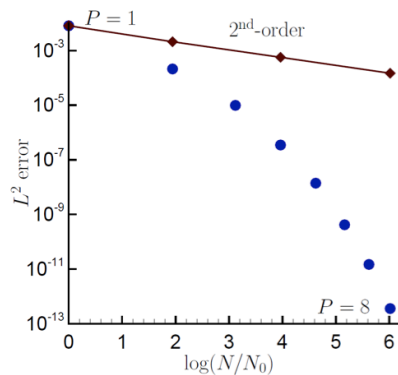
(d) Wedge topology (Wedge6).

Fig. 1. CVFEM element and dual-volume definition for the low-order topologies.

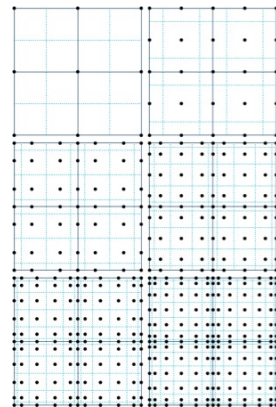


Higher-Order via Polynomial Promotion

- CVFEM (like EBVC and CC) can be viewed as Petrov-Galerkin method
 - Recall how this is defined?
- Basis can also be promoted (linear to quadratic, etc), i.e., *Polynomial Promotion*, Domino, *CTRSP* (2014) as a first example of low-Mach fluids algorithm – or Domino, *JCP* (2018)
- Research Thrust: Possible higher efficiency on NGP due to increased local work)
- However, suitability of higher-order for LES is an open argument – especially when other errors/uncertainties exist



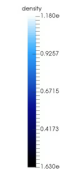
Spectral convergence



Dual-volume for promoted quad4



P=1 (left) and P=4 (right)
Helium plume (VR-density)



Time: 0.055000



Time: 0.055000



Rotating cube (Re 4000,
RPM 3600) P=1 (top) and
P=2 (bottom)